

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- BLACK BORDERS**
- IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- FADED TEXT OR DRAWING**
- BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- SKEWED/SLANTED IMAGES**
- COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- GRAY SCALE DOCUMENTS**
- LINES OR MARKS ON ORIGINAL DOCUMENT**
- REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- OTHER: \_\_\_\_\_**

**IMAGES ARE BEST AVAILABLE COPY.**

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
[www.uspto.gov](http://www.uspto.gov)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/998,511	11/30/2001	Jeremy Alan Arnold	IBM / 194	6560
7590	08/27/2004		EXAMINER PHAM, CHRYSTINE	
Scott A. Stinebruner Wood, Herron & Evans, L.L.P. 2700 Carew Tower 441 Vine Street Cincinnati, OH 45202-2917			ART UNIT 2122	PAPER NUMBER
DATE MAILED: 08/27/2004				

Please find below and/or attached an Office communication concerning this application or proceeding.

<b>Office Action Summary</b>	<b>Application No.</b>	<b>Applicant(s)</b>	
	09/998,511	ARNOLD ET AL.	
	<b>Examiner</b>	<b>Art Unit</b>	
	Chrystine Pham	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --  
**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM  
 THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) Responsive to communication(s) filed on \_\_\_\_\_.
- 2a) This action is **FINAL**.                    2b) This action is non-final.
- 3) Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) Claim(s) 1-30 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) Claim(s) \_\_\_\_\_ is/are allowed.
- 6) Claim(s) 1-30 is/are rejected.
- 7) Claim(s) \_\_\_\_\_ is/are objected to.
- 8) Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) The specification is objected to by the Examiner.
- 10) The drawing(s) filed on 30 November 2001 is/are: a) accepted or b) objected to by the Examiner.  
 Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
 Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
  - a) All
  - b) Some \*
  - c) None of:
    1. Certified copies of the priority documents have been received.
    2. Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
    3. Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1) Notice of References Cited (PTO-892) \*
- 2) Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
 Paper No(s)/Mail Date 1/15/02 \*
- 4) Interview Summary (PTO-413)  
 Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) Notice of Informal Patent Application (PTO-152)
- 6) Other: \_\_\_\_\_.

**DETAILED ACTION**

***Claim Rejections - 35 USC § 102***

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

*A person shall be entitled to a patent unless –*

*(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.*

2. Claims 1, 3-4, 6-10, 13, 15-18, 20, 22-24, 26-30 are rejected under 35 U.S.C. 102(b) as being anticipated by Nishimura et al. (US 5845125), hereinafter, *Nishimura et al.*.

As per claim 1, *Nishimura et al.* teach an apparatus (e.g., see FIG.1 & associated text), a computer-implemented method (e.g., col.1:9-13), a program product stored in a signal bearing medium including transmission medium and recordable medium (e.g., col.9:15-22 & 28-32) for debugging an object-oriented computer program which is resident in a memory (e.g., col.2:3-7, see source code storage section 7 FIG.1 & associated text), the method comprising:

(a) in response to user input, setting an inheritance breakpoint that is associated with a first program entity in the object-oriented computer program in which is identified a method (e.g., see FIG.3 & associated text, col.3:5-6, col.5:42-47, col.14:61-col.15:5); and

(b) halting execution of the object-oriented computer program during debugging in response to reaching an implementation of the method defined in a second program entity in the object-oriented computer program that is different from the first program entity (e.g., col.3:7-14, col.5:48-51, col.14:61-col.15:5, col.16:10-15, see FIG.15 & associated text).

As per claim 3, *Nishimura et al.* teach the computer-implemented method as applied to claim 1, wherein the first program entity is a first class that includes a second implementation of the method, wherein the second program entity is a second class/subclass that inherits from the

first class (e.g., see *derived/child class, base class* col.2:31-33, col.2:21-24, col.12:62-65), and wherein the first implementation of the method in the second class overrides the second implementation of the method in the first class (e.g., col.2:26-31).

As per claim 4, it recites limitations which have been addressed in claim 3, therefore, is rejected for the same reasons as cited in claim 3.

As per claim 6, *Nishimura et al.* teach the computer-implemented method of claim 1, wherein the inheritance breakpoint is additionally associated with the method (e.g., col.13:19-21, col.15:11-14).

As per claim 7, *Nishimura et al.* teach the computer-implemented method of claim 6, wherein setting the inheritance breakpoint includes storing in a breakpoint data structure (e.g., see FIG.5 & associated text) an entry that identifies the first program entity and the method (e.g., see *object identification number & constructor* col.13:29-36, see FIG.11,12 & associated text, col.15:43-49).

As per claim 8, *Nishimura et al.* teach the computer-implemented method of claim 1, further comprising, during loading of a class in the object-oriented computer program, identifying each implementation of the method in the class and setting a breakpoint on such implementation (e.g., FIG.9 & associated text, col.4:21-33, col.15:11-14 & 23-26), wherein halting execution of the object-oriented computer program during debugging in response to reaching the implementation of the method includes reaching a breakpoint set on such implementation (e.g., see FIG.2 & associated text, col.4:34-40, col.16:4-7).

As per claim 9, it recites limitations which have been addressed in claim 8, therefore, is rejected for the same reasons as cited in claim 8.

As per claim 10, *Nishimura et al.* teach the computer-implemented method of claim 9, wherein setting a breakpoint on each implementation of the method includes setting a breakpoint on a first statement in an implementation of the method (e.g., see FIG.30 & associated text, col.1:38-41).

As per claim 13, it recites limitations which have been addressed in claim 3, therefore, is rejected for the same reasons as cited in claim 3.

As per claim 15, *Nishimura et al.* teach a computer-implemented method of debugging an object-oriented computer program, the method comprising:

- (a) receiving user input to halt program execution during debugging in response to reaching any of a plurality of implementations of a method in an object-oriented computer program (e.g., col.6:39-41 & 44-45, col.13:5-10); and
- (b) thereafter setting a breakpoint for at least a subset of the plurality of implementations such that execution of the object-oriented computer program will be halted in response to reaching any of the implementations on which a breakpoint has been set (see claim 9).

As per claim 16, it recites limitations which have been addressed in claims 15 and 1, therefore, is rejected for the same reasons as cited in claims 15 and 1.

As per claims 17, 18, 20, they recite limitations which have been addressed in claims 8, 1, 3 respectively, therefore, are rejected for the same reasons as cited in claims 8, 1, 3.

As per claims 22-24, they recite limitations which have been addressed in claims 7, 9, 8 respectively, therefore, are rejected for the same reasons as cited in claims 7, 9, 8.

As per claims 26-30, they recite limitations which have been addressed in claims 1, 3, 8, and 15, therefore, are rejected for the same reasons as cited in claims 1, 3, 8, and 15.

***Claim Rejections - 35 USC § 103***

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

*(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.*

4. Claims 2, 5, 14, 19, 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Nishimura et al.* in further view of *Coplien et al.* (US 5093914), hereinafter, *Coplien et al.*.

As per claim 2, *Nishimura et al.* teach the computer-implemented method and apparatus as applied to claim 1 above. *Nishimura et al.* do not expressly disclose the first program entity as an interface that identifies the method, and wherein the second program entity is a class that implements the method. However, *Coplien et al.* disclose an apparatus (e.g., see FIG.1,2 & associated text) and a method for debugging an object-oriented computer program (e.g., see Abstract, see FIG.10,11 & associated text) comprising a first program entity which is an abstract class (e.g., col.6:26-35, col.8:54-56, col.8:62-col.9:17, col.9:34-45) or an interface that identifies the method (e.g., see class *Window* FIG.5 & associated text) and a second program entity which is a class that implements the method (e.g., see class *Xwindow*, class *SunviewWindow* FIG.5 & associated text, col.9: 34-45). *Coplien et al.* further disclose setting a breakpoint in a function of the first program entity and halting execution during debugging in response to reaching an implementation of the method defined in a second program entity that is different from the first program entity (e.g., col.8:45-53). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *Coplien et al.* into that of *Nishimura et al.* to replace the first program entity with an object-oriented interface or

abstract class which would produce the expected result with reasonable success. And the motivation for doing so would have been that since interfaces and abstract classes are designed for inheritance which enables data/variables and methods from the base/superclass (i.e., abstract or interface) to be automatically inherited by the derived/subclass without having to define the same data/variables and methods in the subclass, thus minimizing the amount of coding required in implementing the subclass, and in the case of debugging the object-oriented classes, said inheritance feature eliminates the need to manually setting a breakpoint in all implementations (e.g., in the subclass) of the methods identified in the superclass.

As per claims 5, 14, 19, 21, they recite limitations which have been addressed in claim 2, therefore, are rejected for the same reasons as cited in claim 2.

5. Claim 11 is rejected under 35 U.S.C. 103(a) as being unpatentable over *Nishimura et al.* as applied to claim 1 above, in further view of West (US 5740440), hereinafter, *West*.

As per claim 11, *Nishimura et al.* teach the computer-implemented method of claim 9. *Nishimura et al.* do not expressly disclose setting a breakpoint on a method call to an implementation of the method. However, *West* discloses a method of debugging an object-oriented program (e.g., see Abstract) wherein breakpoints are set on method calls (e.g., col.4:1-6). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *West* into that of *Nishimura et al.* to enable setting breakpoint on method calls with reasonable success. And the motivation for doing so would have been that setting breakpoints on a method calls enables the state of the program to be examined at the suspension of execution, and determination can be made as to whether an object in which the method call is contained is newly created so that updates can be reported to the debugger for data tracking or other debug operations.

6. Claims 12 and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Nishimura et al.* as applied to claims 1 and 18 above, in further view of Applicant's Admission of Prior Art (hereinafter APA)(see page 2 – Background of the invention).

As per claim 12, *Nishimura et al.* teach the computer-implemented method of claim 1. *Nishimura et al.* do not expressly disclose setting the inheritance breakpoint includes associating a user-specified condition with the inheritance breakpoint, and wherein halting execution of the object-oriented computer program during debugging in response to reaching the implementation of the method is performed only if the user-specified condition has been met. However, APA discloses associating a user-specified condition with the breakpoint , and wherein halting execution of the object-oriented computer program during debugging in response to reaching the implementation of the method is performed only if the user-specified condition has been met (e.g., see *after the breakpoint has been hit X times* pg.2:13-18). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of APA into that of *Nishimura et al.* to obtain conditional (i.e., user-specified condition) breakpoints with reasonable success. And the motivation for doing so would have been determining and halting execution only when the user-specified condition (i.e., breakpoint has been hit X times) for a breakpoint has been met would allow the user of the debugging method more control over program suspension. That is to say, by specifying and monitoring said condition associated with breakpoint A, the user could skip over X-1 breakpoint iteration(s) (which entail X-1 suspensions of program execution) and jump to the breakpoint of interest (i.e., when breakpoint A has been hit X times), thus making program debugging a more effective and efficient process.

As per claim 25, it recites limitations which have been addressed in claim 12, therefore, is rejected for the same reasons as cited in claim 12.

***Conclusion***

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure:
  - o Interception of unit creation requests by an automatic distributed partitioning system, Hunt (US 6629123)
  - o Portable debugging services utilizing a client debugger object and a server debugger object with flexible addressing support, You (US 6158045)
  - o Debugging system with portable debug environment-independent client and non-portable platform-specific server, You et al. (US 5815653)
  - o System and method for distributed debugging and recording medium on which control programs are recorded, Koyama (US 20010004766)
  - o Source level debugger for debugging source programs, Mizumoto et al. (US 6550056)
  - o Debugging optimized code using data change points, Curreri et al. (US 6091896)
  - o Method and apparatus for testing software on a computer network, Cardoza et al. (US 5630049)
  - o Multi-threaded break-point, Bates et al. (US 6681384)
  - o Debugger thread identification points Bates et al. (US 6378125)
  - o Method and apparatus for debugging and tuning a process control network having distributed control functions, Larson et al. (US 6044305)
  - o Method of debugging a computer program, Padawer et al. (US 5124989)
  - o Break points system in a software monitor where entry address of break point routing equals op code of call instruction Haswell-Smith (US 4866665)
  - o Microprocessor system debug tool Boyce et al. (US 4796258)
  
8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chrystine Pham whose telephone number is 703.605.1219. The examiner can normally be reached on Mon-Fri, 8:30am-5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q Dam can be reached on 703.305.4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Chrystine Pham  
Examiner  
2122  
August 20, 2004



**ANTONY NGUYEN-BA  
PRIMARY EXAMINER**